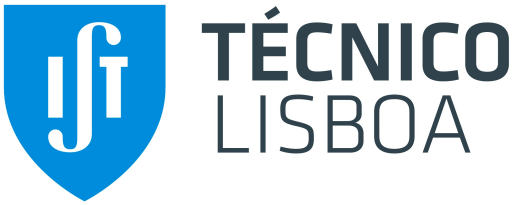


Instituto Superior Técnico

Sistemas de Controlo de Tráfego

Posicionamento de um Recetor em GPS

Mestrado Integrado em Engenharia Aeroespacial

Renato Loureiro, 89708

Santiago Rodríguez, 90360

2020/21

Índice

1 Introdução

No âmbito da unidade curricular de Sistemas de Controlo de Tráfego foi nos proposto a elaboração de um projeto sobre o posicionamento de um recetor em *Global Positioning System* (GPS). O projeto consiste em implementar a simulação da trajetória. Inicialmente, gerar-se-á uma constelação de satélites GPS para um tempo predefinido utilizando o almanaque do website Celestrak.com. A seguir, serão determinados os satelites à vista numa determinada localização sobre a superficie da Terra. Posteriormente, determinar-se-á a subconstelação ótima para um certo número de satelites à vista (maior ou igual a quatro) utilizando a minimização do parâmetro PDOP. Subsequentemente, implementar-se-ão os algoritmos dos mínimos quadrados e o filtro de Kalman generalizado (EKF) utilizando pseudo-distâncias com imperfeições atmosféricas como observações. Estes métodos irão estimar as posição do recetor (latitude, longitude e altitude) e a sua velocidade. Finalmente, analisar-se-ão os resultados obtidos e comparar-se-á o desempenho de ambos algoritmos.

É de salientar que o modelo da dinâmica usado neste trabalho é o modelo PV, embora está presente nos anexos um estudo realizado com PVA.

2 Sistemas de coordenadas

De modo a elaborar cálculos entre as grandezas dos satélites transmissores e o recetor é crucial que ambos se encontrem no mesmo referencial e com o mesmo tipo de coordenadas. Para isto é preciso realizar a conversão de coordenadas e referenciais. A posição de cada satelite é determinada utilizando coordenadas inerciais no referencial *Earth-Centered Inertial Coordinates* (ECI). Este referencial carateriza-se por estar centrado na Terra onde o plano xy coincide com o plano equatorial desta. Relativamente aos eixos, o eixo xx mantém-se permanentemente fixo apontando para uma determinada posição da esfera celeste (equinócio vernal), o eixo zz aponta para o pólo norte e o eixo yy é escolhido de forma a gerar um sistema de coordenadas destro.

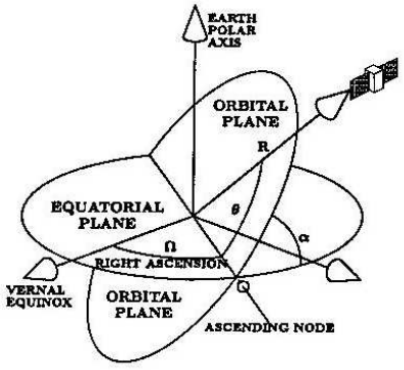


Figura 1: Coordenadas do satélite e referencial ECI

As órbitas dos satélites são determinadas no referencial ECI e depois são convertidas em co-

ordenadas do referencial *Earth Centered Earth Fixed* (ECEF). Este sistema também se encontra centrado e fixo na Terra, mas apresenta uma diferença relativamente aos eixos comparando-o com o sistema ECI. O eixo xx aponta na direção de longitude de 0º, o eixo yy na direção de longitude de 90ºE e o eixo zz é selecionado de modo a formar um sistema destro. As posições dos satélites são enviadas para o recetor na mensagem de navegação utilizando coordenadas ECEF. A partir da informação do almanaque, a posição do satelite GPS em coordenadas ECEF pode ser aproximada pela seguinte expressão

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R \begin{bmatrix} \cos(\theta)\cos(\Omega) - \sin(\theta)\sin(\Omega)cos(\alpha) \\ \cos(\theta)\sin(\Omega) + \sin(\theta)\cos(\Omega)cos(\alpha) \\ \sin(\theta)\sin(\alpha) \end{bmatrix} \tag{1}$$

Em que R corresponde ao raio da orbita, Ω a longitude ECI onde o plano orbital interseta o plano equatorial quando o satelite passa do hemisfério sul para o norte, θ a posição do satelite no plano da orbita (medida a partir do plano equatorial) e α a inclinação do plano orbital relativamente ao plano equatorial. A Figura 1 ilustra de forma gráfica a definição destas variáveis no referencial ECI.

Uma vez que a equação de navegação é geralmente resolvida em coordenadas ECEF é usual transformar as coordenadas ECEF do recetor em latitude, longitude e altitude, e viceversa. De modo a realizar esta transformação é fundamental definir um modelo físico da Terra. Neste caso o modelo físico utilizado é o *World Geodesic System 1984* (WGS-84). Neste trabalho utilizou-se a biblioteca *Mapping Toolbox* de modo a realizar estas transformações.

A trajetória da aeronave proporcionada no enunciado encontra-se representada num referencial *East-North-Up* (ENU). Sendo assim, é necessário encontrar uma relação entre este referencial local e o referencial ECEF. O sistema de coordenadas ENU carateriza-se por apresentar os seus eixos xx e yy tangentes aos paralelos e meridianos, respetivamente. Estes dois eixos formam um plano tangente à superficie da Terra. Este sistema encontra-se centrado num ponto genérico $P_u = (x_u, y_u, z_u)$, no caso deste projeto será o ponto A, especificado no enunciado. Para transformar as coordenadas ECEF para ENU utiliza-se a fórmula

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} -\sin(\lambda_u) & \cos(\lambda_u) & 0 \\ -\sin(\phi_u)\cos(\lambda_u) & -\sin(\phi_u)\sin(\lambda_u) & \cos(\phi_u) \\ \cos(\phi_u)\cos(\lambda_u) & \cos(\phi_u)\sin(\lambda_u) & \sin(\phi_u) \end{bmatrix} \begin{bmatrix} x - x_u \\ y - y_u \\ z - z_u \end{bmatrix} \tag{2}$$

Em que $P = (x', y', z')$ são as coordenadas de um ponto no referencial ENU, $P = (x, y, z)$ as coordenadas de um ponto no referencial ECEF e ϕ_u e λ_u correspondem à latitude e longitude de P_u respetivamente. Verifica-se que para $z'(t) > 0$ o ponto P situa-se acima do horizonte local, sendo assim a condição para calcular o número de satélites à vista em qualquer ponto da superfície da Terra (N_{view}).

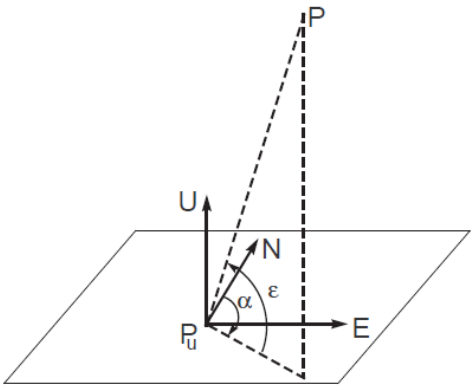


Figura 2: Definição do referencial local ENU

Analisando a figura 2, pode-se obter o ângulo de azimute (α) e o ângulo de elevação (ε) do ponto P a partir de

$$\tan(\alpha) = \frac{x'}{y'} \iff \alpha = arctan2(x', y') \tag{3}$$

Em que

$$arctan2(x', y') = \begin{cases} arctan(x'/y'), & x > 0 \\ arctan(x'/y') + \pi, & x < 0, y \geq 0 \\ arctan(x'/y') - \pi, & x < 0, y < 0 \end{cases} \tag{4}$$

E

$$\sin(\varepsilon) = \frac{z'}{\sqrt{(x')^2 + (y')^2 + (z')^2}} \iff \varepsilon = \arcsin\left(\frac{z'}{\sqrt{(x')^2 + (y')^2 + (z')^2}}\right) \tag{5}$$

3 Medições do sistema GPS

De forma a determinar a posição do recetor (x_u, y_u, z_u) e o desvio de relógio t_u do recetor relativamente ao sistema, são efetuadas medições das pseudo-distâncias associadas aos N satélites considerados ($N \geq 4$) resultando na seguinte expressão para o cálculo da pseudo-distância ρ_i para cada um dos satélites (X_i, Y_i, Z_i).

$$\rho_i = \sqrt{(X_i - x_u)^2 + (Y_i - y_u)^2 + (Z_i - z_u)^2} + ct_u + n_i \tag{6}$$

em que ρ_i representa a pseudo-distância, (X_i, Y_i, Z_i) designa a posição do satelite i e n_i é o erro associado à medição ρ_i devido ao cancelamento incompleto dos erros introduzidos pela ionosfera/troposfera, ruido térmico, etc. Geralmente os erros são considerados gaussianos, independentes, de média nula e com variâncias iguais.

Para linearizar a equação de navegação, a expressão em (6) pode ser definida como

$$\rho_i = \sqrt{(X_i - x_u)^2 + (Y_i - y_u)^2 + (Z_i - z_u)^2} + b_u, \quad i = 1, ..., N \tag{7}$$

em que o erro n_i é desprezado e $b_u = ct_u$. O respetivo modelo incremental é

$$\Delta\rho_i = \frac{\partial\rho_i}{\partial x_u}\Delta x_u + \frac{\partial\rho_i}{\partial y_u}\Delta y_u + \frac{\partial\rho_i}{\partial z_u}\Delta z_u + \frac{\partial\rho_i}{\partial b_u}\Delta b_u. \tag{8}$$

Esta equação também pode ser representada de forma matricial

$$\Delta\rho = G\Delta X \tag{9}$$

em que

$$\Delta\rho_i = \begin{bmatrix}\Delta\rho_1 \\ ... \\ \Delta\rho_N\end{bmatrix}, \quad \Delta X = \begin{bmatrix}\Delta x_u \\ \Delta y_u \\ \Delta z_u \\ \Delta b_u\end{bmatrix} \tag{10}$$

e

$$G = \begin{bmatrix}\frac{\partial\rho_1}{\partial x_u} & \frac{\partial\rho_1}{\partial y_u} & \frac{\partial\rho_1}{\partial z_u} & 1 \\ ... & ... & ... & ... \\ \frac{\partial\rho_N}{\partial x_u} & \frac{\partial\rho_N}{\partial y_u} & \frac{\partial\rho_N}{\partial z_u} & 1\end{bmatrix} \tag{11}$$

Sendo que a matriz G é denominada por matriz geométrica. A partir da definição de ρ_i em 7, pode-se calcular as derivadas seguintes

$$\frac{\partial\rho_i}{\partial x_u} = -\frac{X_i - x_u}{D_i}, \quad \frac{\partial\rho_i}{\partial y_u} = -\frac{Y_i - y_u}{D_i}, \quad \frac{\partial\rho_i}{\partial z_u} = -\frac{Z_i - z_u}{D_i}, \quad \frac{\partial\rho_i}{\partial b_u} = 1 \tag{12}$$

onde $D_i = \sqrt{(X_i - x_u)^2 + (Y_i - y_u)^2 + (Z_i - z_u)^2}$. O método dos mínimos quadrados pode ser utilizado para resolver a equação 9 obtendo a seguinte solução

$$\Delta X = (G^T G)^{-1} G^T \Delta\rho, \quad N \geq 4. \tag{13}$$

Sabendo que $\Delta\rho$ e ΔX representam os os erros das pseudo-distâncias e de posição (incluindo o erro de desvio de relógio do recetor), respetivamente, a equação (13) caracteriza uma relação linear entre ambos erros. Os erros das pseudo-distâncias são consideradas conjuntamente como variáveis aleatórias, independentes, gaussianas de média nula. Seguindo esta lógica, assumindo uma geometria fixa, ΔX também é um vetor gaussiano de média nula com uma matriz de covariância

$$\begin{aligned} cov(\Delta X) &= E\{\Delta X \Delta X\} \\ &= E\{(G^T G)^{-1} G^T \Delta\rho (\Delta\rho)^T G (G^T G)^{-1}\} \\ &= (G^T G)^{-1} G^T E\{\Delta\rho (\Delta\rho)^T\} G (G^T G)^{-1} \end{aligned} \tag{14}$$

Sendo que $G(G^T G)^{-1}$ é uma matriz simétrica.

Admitindo que as componentes $\Delta\rho$ são identicamente distribuídas e independentes com uma variância igual ao quadrado do parâmetro UERE (comum aos vários satélites). Obtendo

$$E\{\Delta\rho(\Delta\rho)^T\} = I_{n\times n}\sigma_{UERE}. \tag{15}$$

Aplicando esta expressão em (14) obtém-se

$$cov\{\Delta X\} = \sigma^2_{UERE}(G^TG)^{-1} \tag{16}$$

As componentes da matriz $(G^TG)^{-1}$ quantificam a forma como os erros das pseudo-distâncias são convertidos em erros de ΔX. A covariancia de ΔX é dada por

$$cov\{\Delta X\} = \begin{bmatrix} \sigma^2_{x_u} & \sigma^2_{x_uy_u} & \sigma^2_{x_uz_u} & \sigma^2_{x_ub_u} \\ \sigma^2_{x_uy_u} & \sigma^2_{y_u} & \sigma^2_{y_uz_u} & \sigma^2_{y_ub_u} \\ \sigma^2_{x_uz_u} & \sigma^2_{y_uz_u} & \sigma^2_{z_u} & \sigma^2_{z_ub_u} \\ \sigma^2_{x_ub_u} & \sigma^2_{y_ub_u} & \sigma^2_{z_ub_u} & \sigma^2_{b_u} \end{bmatrix} \tag{17}$$

Os parâmetros *Dilution Of Precision* (DOP) em GPS são definidos em termos de quociente das combinações das componentes da matriz cov(ΔX) e do parâmetro UERE e são utilizados para caraterizar a precisão das várias componentes da solução posição/tempo como a posição ou o tempo. O parâmetro mais geral designa-se por Diluição da Precisão Geométrica (GDOP). O GDOP é um fator de ampliação que depende exclusivamente da geometria recetor/satélites, este pode ser definido por

$$GDOP = \frac{1}{\sigma_{UERE}}\sqrt{\sigma^2_{x_u} + \sigma^2_{y_u} + \sigma^2_{z_u} + \sigma^2_{b_u}} \tag{18}$$

A expressão (18) mostra que o *Root-Mean-Square Error* (erro rms) da posição (incluindo a contribuição do desvio do relógio do recetor) é $\sqrt{\sigma^2_{x_u} + \sigma^2_{y_u} + \sigma^2_{z_u} + \sigma^2_{b_u}} = \sigma_{UERE}GDOP$, em que σ_{UERE} depende das características do recetor, do satélite e outros segmentos como é ilustrado na Tabela 1.

Tabela 1: Erros típicos das pseudo-distâncias para o código GPS C/A

Segmento	Fonte de Erro	Error de GPS 1σ (m)
Espaço	estabilidade do relógio de satélite	1.1
	atraso de grupo de L1	0.3
Controlo	erros de efemérides	0.8
Utilizador	atraso ionosférico	7.0
	atraso troposférico	0.2
	ruído do recetor e resolução	0.1
	multipercurso	0.2
UERE	total	7.1

O parâmetro GDOP também pode ser definido como

$$GDOP = \sqrt{tr((G^TG)^{-1})} \tag{19}$$

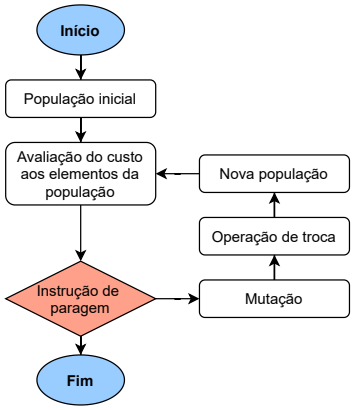
4 Minimização do PDOP

Como foi referido no capítulo anterior os parâmetros DOP são usados para caraterizar a precisão das componentes da solução posição/tempo, no presente estudo de simulação centrou-se apenas na averiguação da diluição de precisão da posição (PDOP), sendo este parâmetro calculado em termos das componentes de $(H^T H)^{-1}$ expresso na seguinte equação

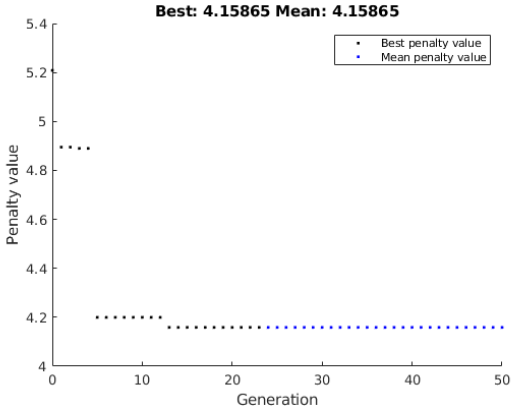
$$PDOP = \sqrt{h_{11} + h_{22} + h_{33}} \tag{20}$$

Valores baixos de PDOP corresponde a um bom arranjo da geometria dos satélites/recetor. Deste modo, o problema da diluição apresenta-se como um problema de otimização (minimização) discreta onde a função de custo associada é a expressão de cálculo do PDOP presente na equação (20), ou seja, o problema consiste em determinar o conjunto de satélites que minimiza o PDOP.

O algoritmo de otimização implementado consiste num algoritmo genético tal como o apresentado no fluxograma da Figura 3a onde para além de determinar o melhor conjunto de satélites que minimiza a função de custo imposta, também permite colocar restrições relativamente ao número de satélites a utilizar e também quais são considerados como visíveis e que podem estar inseridos na otimização.



(a) Fluxograma do algoritmo genético implementado no problema de minimização do PDOP.



(b) Resultados obtidos do algoritmo genético após 50 gerações com a restrição de utilização de apenas 4 satélites - cada geração considerada detém 100 combinações.

Figura 3: Resultados do algoritmo genético no problema de minimização do PDOP.

Para um caso onde o número de satélites visíveis é 15, corresponde a um número total de combinações possíveis de 31616, sendo que apenas é considerado combinações com 4 ou mais satélites - verifica-se que a simples abordagem de calcular o custo para cada combinação possível detém um peso computacional considerável e é por esta razão que o algoritmo genético é vantajoso, pois não calcula para todas as combinações, embora detém o problema de convergir

para um mínimo local. Por experiência de simulação o valor de PDOP final é muito semelhante caso seja mínimo local ou o mínimo global.

Verifica-se pelo gráfico presente na Figura 3b que em apenas 20 gerações o algoritmo converge tendo neste caso em particular obtido um valor mínimo de PDOP igual a 4.15865.

5 Modelo da dinâmica

O filtro de Kalman generalizado (EKF) implementado depende do modelo da dinâmica adotado, dado que dependendo da escolha do modelo, o número de variáveis de estado x_k do sistema é diferente, mas tambem o comportamento e performance do mesmo. No presente estudo como o recetor não está fixo no espaço, a utilização de um modelo simples P da dinâmica implicaria valores de erros elevados, assim sendo decidiu-se utilizar o modelo PV - o que corresponde a um vetor de estado x_k de dimensão 8 - 6 relativamente à posição e velocidade em cada um dos eixos x, y, z e 2 devido à dinâmica de relógio.

5.1 Descrição do modelo PV

Seja o vetor de estado $x_k = [x_{1,k}, ..., x_{8,k}]^T$, em que $x_{1,k}$ e $x_{2,k}$ são, respetivamente, a posição e a velocidade relativamente à coordenada x_u do recetor, $x_{3,k}$ e $x_{4,k}$ representam a posição e a velocidade relativamente à coordenada y_u, etc. O modelo da dinâmica discreto no tempo é dado por

$$\begin{bmatrix} x_{1,k+1} \\ x_{2,k+1} \\ x_{3,k+1} \\ x_{4,k+1} \\ x_{5,k+1} \\ x_{6,k+1} \\ x_{7,k+1} \\ x_{8,k+1} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & & & & & & \\ & 1 & & & & & 0 & \\ & & 1 & \Delta t & & & & \\ & & & 1 & & & & \\ & & & & 1 & \Delta t & & \\ & & & & & 1 & & \\ & & 0 & & & & 1 & \Delta t \\ & & & & & & & 1 \end{bmatrix} \begin{bmatrix} x_{1,k} \\ x_{2,k} \\ x_{3,k} \\ x_{4,k} \\ x_{5,k} \\ x_{6,k} \\ x_{7,k} \\ x_{8,k} \end{bmatrix} + \begin{bmatrix} u_{1,k} \\ u_{2,k} \\ u_{3,k} \\ u_{4,k} \\ u_{5,k} \\ u_{6,k} \\ u_{7,k} \\ u_{8,k} \end{bmatrix}$$

em que os elementos da matriz de covariância do ruído Q_k são

$$q_{i,i} = q_v \frac{(\Delta t)^3}{3} \qquad q_{i,i+1} = q_{i+1,i} = q_v \frac{(\Delta t)^2}{2}$$

$$q_{i+1,i+1} = q_v \Delta t \text{ com: } i = 1, 3, 5 \qquad q_{7,7} = q_\phi \Delta t + \frac{q_f (\Delta t)^3}{3}$$

$$q_{7,8} = q_{8,7} = \frac{q_f (\Delta t)^2}{2} \qquad q_{8,8} = q_f \Delta t$$

sendo os restantes elementos de Q_k nulos.

O parâmetro q_v pode ser calculado recorrendo à aproximação

$$\begin{aligned} aceleração &\approx \frac{x_{i,k+1} - x_{i,k}}{\Delta t} = \frac{u_{i,k}}{\Delta t} \\ q_v &= E\{u_{i,k}^2\} \geq \Delta t^2 (aceleração)^2 \ , \quad i = 2, 4, 6 \end{aligned} \tag{21}$$

5.2 Modelo de estado do relógio do recetor

O modelo de relógio implementado é definido por um vector de estado de dimensão 2 em que tanto a frequência como a fase apresentam variações do tipo movimento browniano em intervalos razoáveis de tempo. A Figura 4 mostra o modelo de estado considerado na equação (23).

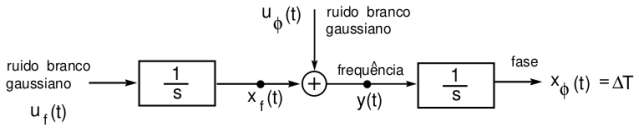


Figura 4: Representação do modelo de estado do relógio do recetor.

Os ruídos brancos $u_\phi(t)$ e $u_f(t)$ são independentes, têm médias nulas e são caraterizados pela matriz de covariância presente na equação (22) em que q_f e q_ϕ são respetivamente as densidades espectrais de potência de $u_f(t)$ e de $u_\phi(t)$.

$$Q_u = \begin{bmatrix} q_\phi & 0 \\ 0 & q_f \end{bmatrix} \tag{22}$$

A equação da dinâmica que carateriza o modelo de estado do relógio do recetor é

$$\begin{bmatrix} \dot{x}_\phi(t) \\ \dot{x}_f(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_\phi(t) \\ x_f(t) \end{bmatrix} + \begin{bmatrix} u_\phi(t) \\ u_f(t) \end{bmatrix} \tag{23}$$

A equação do modelo discreto no tempo correspondente é

$$\begin{bmatrix} x_{\phi,k+1} \\ x_{f,k+1} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{\phi,k} \\ x_{f,k} \end{bmatrix} + \begin{bmatrix} u_{\phi,k} \\ u_{f,k} \end{bmatrix} \tag{24}$$

onde a matriz de covariância de ruído correspondente é

$$Q_k = \begin{bmatrix} q_\phi \Delta t + \frac{q_f (\Delta t)^3}{3} & \frac{q_f (\Delta t)^2}{2} \\ \frac{q_f (\Delta t)^2}{2} & q_f \Delta t \end{bmatrix} \tag{25}$$

$$q_\phi \approx \frac{h_0}{2} \qquad q_f \approx 2\pi^2 h_{-2} \tag{26}$$

Os parâmetros h_0 e h_{-2} admitem certos valores dependendo do tipo de relógio como está apresentado na Tabela 2 - no caso em estudo é utilizado um oscilador de cristal compensado em temperatura.

Tabela 2: Valores típicos de h_0 e h_{-2} para diferentes tipos de relógios usados em recetores GPS.

tipo de oscilador	h_0	h_{-2}
cristal compensado em temperatura	$2 \cdot 10^{-19}$	$2 \cdot 10^{-20}$
cristal em forno	$8 \cdot 10^{-20}$	$4 \cdot 10^{-23}$
rubídio	$2 \cdot 10^{-20}$	$4 \cdot 10^{-29}$

É de salientar que a implementação da anterior matriz de covariância de ruído com os valores presentes na tabela anterior deve cada entrada da matriz ser multiplicada por $c^2 = 9 \cdot 10^{16}$.

6 Algoritmo dos mínimos quadrados

Para resolver esta equação vetorial de navegação pode ser aplicado o método dos mínimos quadrados, também designado por *Least Mean Squares* (LMS). Como foi verificado na secção 3, a solução do sistema linear em (9) é definida pela expressão (13). Mesmo assim, ao linearizar o sistema são introduzidos erros nas estimativas, de modo a que é necessário utilizar um método iterativo para melhora-las.

O processo iterativo aplicado é o algoritmo dos mínimos quadrados, cujo procedimento é ilustrado na Figura 5. Inicialmente sabe-se as medidas das pseudo-distâncias (R_i), as coordenadas dos M transmissores (X_i, Y_i, Z_i) e a posição inicial do recetor $\hat{P}_0 = (\hat{x_0}, \hat{y_0}, \hat{z_0}, \hat{b_0})$. De seguida, calculam-se as estimativas da matriz geométrica ($\hat{G}$) e das pseudo-distâncias ($\hat{R}_i$) a partir das estimativas de posição do recetor e dos satélites e das expressões (7) e (11) . Posteriormente, calcula-se o erro das pseudo-distâncias. Este erro será utilizado para calcular o erro de posição (ΔX) com ajuda de $\hat{G}$. Finalmente, a estimativa da posição do recetor é atualizada somando a estimativa atual com o ΔX. O processo é repetido até que a nova estimativa de posição seja igual à estimativa anterior. É importante mencionar que neste trabalho a estimativa da posição inicial utilizada é a posição do ponto A da trajetoria referida no enunciado.

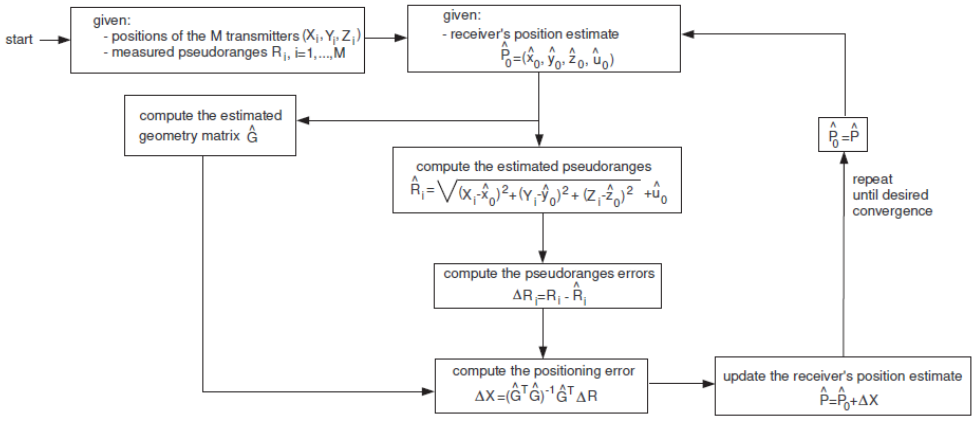


Figura 5: Fluxograma do algoritmo dos mínimos quadrados aplicado na resolução da equação de navegação em GPS

Por efeitos de análise e comparação de algoritmos, é necessário determinar a velocidade do recetor utilizando o LMS, mas o modelo descrito a cima não fornece uma estimativa para a velocidade. Deste modo recorre-se a uma aproximação de primeira ordem para os valores da velocidade embora este processo resulta em erros de estimação elevados. As diferenças finitas utilizadas estão presentes nas equações seguintes onde é de salientar que nos limites é utilizado diferenças finitas progressivas/regressivas (27) e que no interior é aplicado diferenças finitas centrais (28).

$$v_i(t) = \frac{p_i(t) - p_i(t - \Delta t)}{\Delta t} \qquad v_i(t) = \frac{p_i(t + \Delta t) - p_i(t)}{\Delta t} \tag{27}$$

$$v_i(t) = \frac{p_i(t + \Delta t) - p_i(t - \Delta t)}{2\Delta t}\ , \quad i = x, y, z \tag{28}$$

7 Filtro de Kalman

Como alternativa ao método mínimos quadrados, utilizou-se o filtro de Kalman na resolução da equação vetorial de navegação. Escolheu-se um filtro de Kalman generalizado visto que apesar das equações do sistema dinâmico sejam lineares (independentemente do modelo de dinâmica escolhido), a equação das observações é não linear, tornando o *Extended Kalman Filter* (EKF) a opção mais adequada.

As observações são incorporadas em intervalos de tempo discreto, neste caso de 0.5s (f = 2Hz), e o modelo das observações é linearizado relativamente à melhor estimativa corrente do estado. O modelo não linearizado das observações é representado por

$$z_k = h[x_k] + v_k \tag{29}$$

Aplicando este modelo das observações ao problema observa-se que o vetor das observações se baseia nas pseudo-distâncias medidas $z_k = [\rho_{1,k}...\rho_{n,k}]^T$, com $n \geq 4$,em que n representa o número de satélites. Relativamente à função $h[x]$

$$h[x] = \begin{bmatrix} \sqrt{(X_1 - x_1)^2 + (Y_1 - x_3)^2 + (Z_1 - x_5)^2} + x_7 \\ ... \\ \sqrt{(X_n - x_1)^2 + (Y_n - x_3)^2 + (Z_n - x_5)^2} + x_7 \end{bmatrix} \tag{30}$$

em que X_i, Y_i, Z_i são as coordenadas do satélite i, com $i = 1, ..., n$ e x_1, x_3, x_5 e x_7 correspondem a x_u, y_u, z_u e b_u, respetivamente.

No que se refere a v_k, este apresenta uma matriz de covariância diagonal definida por

$$R_k = \begin{bmatrix} \sigma^2_{1,UERE} & 0 & ... & 0 \\ 0 & \sigma^2_{2,UERE} & & 0 \\ ... & & ... & \\ 0 & 0 & & \sigma^2_{n,UERE} \end{bmatrix} \tag{31}$$

Neste caso definiram-se todas as variâncias iguais a σ^2_{UERE}, logo $R_k = \sigma^2_{UERE} I$. A matriz das observações do EKF é definida por

$$H_k = \left[\frac{\partial h_i[x(k|k-1)]}{\partial x_j}\right]_{n\times l} \tag{32}$$

onde l é a dimensão do vetor de estado. Para o modelo PV, as derivadas em função das posições (estados x_1, x_3, x_5 e x_7) serão iguais às que foram apresentadas em (12). As derivadas em função das velocidades (estados x_2, x_4, x_6 e x_8) serão nukas dado que a equação das pseudo-distâncias não dependem destas variáveis. Deste modo

$$H_k = -\begin{bmatrix} \frac{X_1-\hat{x_u}}{r_i} & 0 & \frac{Y_1-\hat{y_u}}{r_i} & 0 & \frac{Z_1-\hat{z_u}}{r_i} & 0 & -1 & 0 \\ \dots & & & \dots & & & & \dots \\ \frac{X_n-\hat{x_u}}{r_i} & 0 & \frac{Y_n-\hat{y_u}}{r_i} & 0 & \frac{Z_n-\hat{z_u}}{r_i} & 0 & -1 & 0 \end{bmatrix} \tag{33}$$

Em que

$$\begin{bmatrix}\hat{x_u} & \hat{y_u} & \hat{z_u}\end{bmatrix}^T = \begin{bmatrix}\hat{x_1}(k|k-1) & \hat{x_3}(k|k-1) & \hat{x_5}(k|k-1)\end{bmatrix} \tag{34}$$

O processo de resolução de navegação elaborado pelo EKF pode ser simplificado pela Figura 6, em que temos os passos de filtragem e predição que compõem este processo recursivo. No fluxograma ilustrado considerou-se por facilidade que

$$\hat{x}_k^- = \hat{x}(k|k-1), \quad \hat{x}_k^+ = \hat{x}(k|k), \quad P_k^+ = P(k|k), \quad P_{k+1}^- = P(k+1|k), \quad \hat{z}_k = h[\hat{x}(k|k-1)]$$

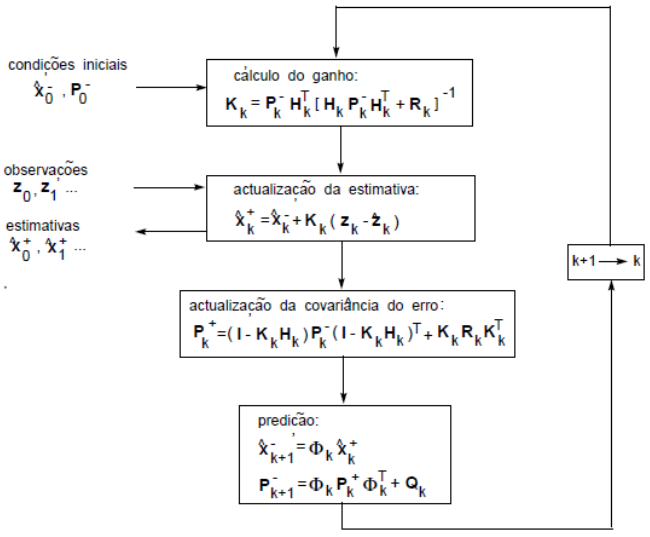


Figura 6: Fluxograma de filtro de Kalman generalizado aplicado na resolução da equação de navegação em GPS [1]

É de salientar que as condições iniciais utilizadas neste projeto são as coordenadas do ponto A da trajetoria relativamente à posição e uma velocidade inicial nula.

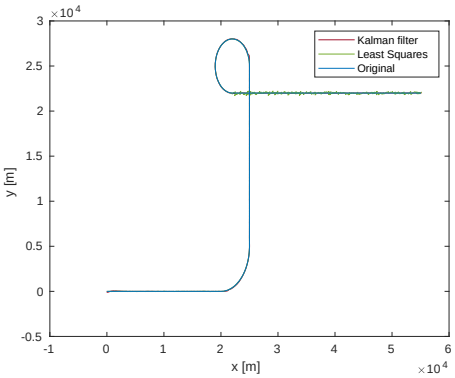
8 Resultados

Na elaboração do EKF recorrendo ao modelo PV, considerou-se dois casos distintos: um caso onde a matriz da covariância Q_k é estática, ou seja, q_v é uma constante ao longo de toda a simulação que é definida pelo utilizador à priori, e outro caso onde a matriz da covariância Q_k não é estática, ou seja, q_v varia segundo a lei expressa na equação (21). Estas duas situações são interessantes na medida que ambas apresentam vantagens e desvantagens distintas - quando Q_k é estático evita ter de se estimar q_v em cada iteração, permitindo menos peso computacional no Kalman Filter, mas leva o sistema a ter um erro de seguimento em situações de aceleração.*

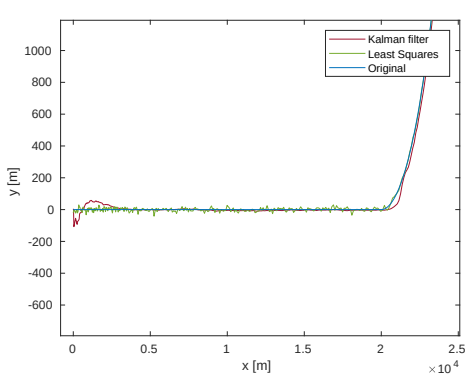
Nas secções subsequentes da análise de resultados, estar-se-á a considerar a matriz Q_k variável do modelo PV e que segue a lei da equação (21).

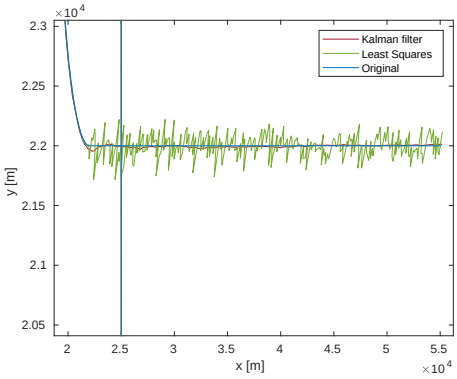
No estudo da performance dos dois algoritmos aqui apresentados, respetivamente o EKF e o LMS, avaliou-se a capacidade de cada um destes permitir fazer um seguimento correto da posição e da velocidade do recetor - na Figura 7 está apresentado os resultados obtidos da trajetória percorrida pelo recetor determinado pelos dois algoritmos, mas também está representado a trajetória original do recetor.

*Como esta observação não tem um papel determinante no estudo deste trabalho, os resultados referentes estarão presentes no **Anexo**



(a) Trajetória total da simulação





(b) Zoom da trajetória no caminho AB (c) Zoom da trajetória no caminho EF

Figura 7: Comparação do comportamento do LMS e do EKF

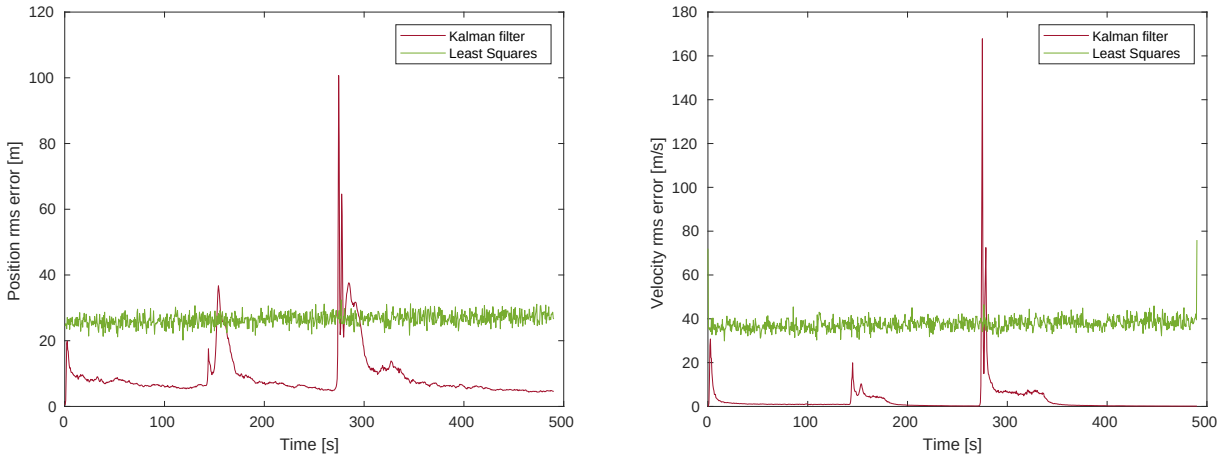
Nos resultados anteriores, está incluído a presença de ruído não gaussiano devido à ionosfera e também a presença da situação de canyon[†] no caminho EF. Considerou-se também na simulação anterior que a otimização do PDOP poderia ser realizada sem qualquer restrição do número de satélites visíveis a utilizar em toda a trajetória exceto no caminho EF.

Verifica-se que o EKF é mais robusto à presença de ruído, mas em consequência dessa capacidade, ele é menos reativo a variações de estado do sistema, observando-se um delay comparativamente ao LMS, por exemplo na zona de transição entre o caminho AB e BC, mas também entre o caminho DE e EF.

Constata-se que o LMS para além de ser menos robusto que o EKF, na situação de canyon, onde o PDOP apresenta um valor consideravelmente superior ao utilizado no resto da trajetória, a precisão do LMS é muito afetada comparativamente ao EKF.

Os resultados anteriormente mencionados são agora complementados pelas Figuras 8, 9, 10 e 11, onde está apresentado o erro rms da posição/velocidade sem consideração do canyon, um exemplo de dados da velocidade v_x, v_y e v_z, efeito do canyon no EKF e efeito do canyon no LMS, respetivamente.

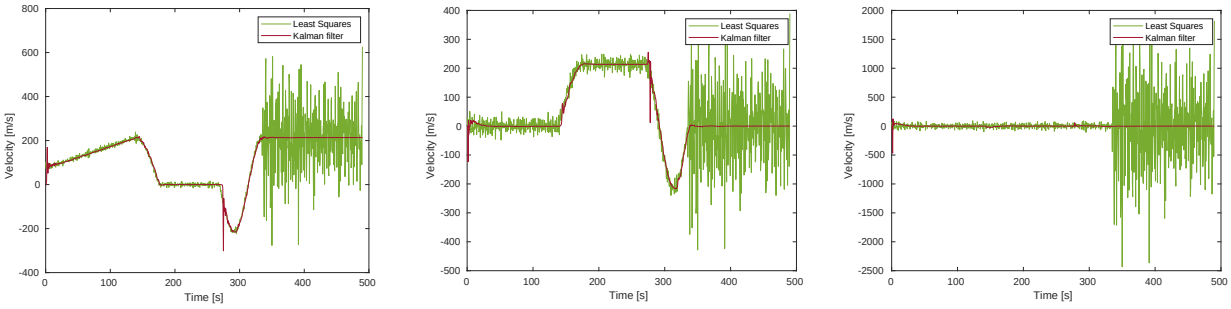
[†]A situação de canyon é definida como a utilização de apenas 4 satélites visíveis sendo estes os que apresentam maior inclinação relativamente ao plano horizontal do recetor



(a) Erro rms da posição (b) Erro rms da velocidade

Figura 8: Erro rms da posição e velocidade ao longo da trajetória pelo LMS e EKF. Estes resultados não têm o canyon no caminho EF dado que essa análise será feita em separado.

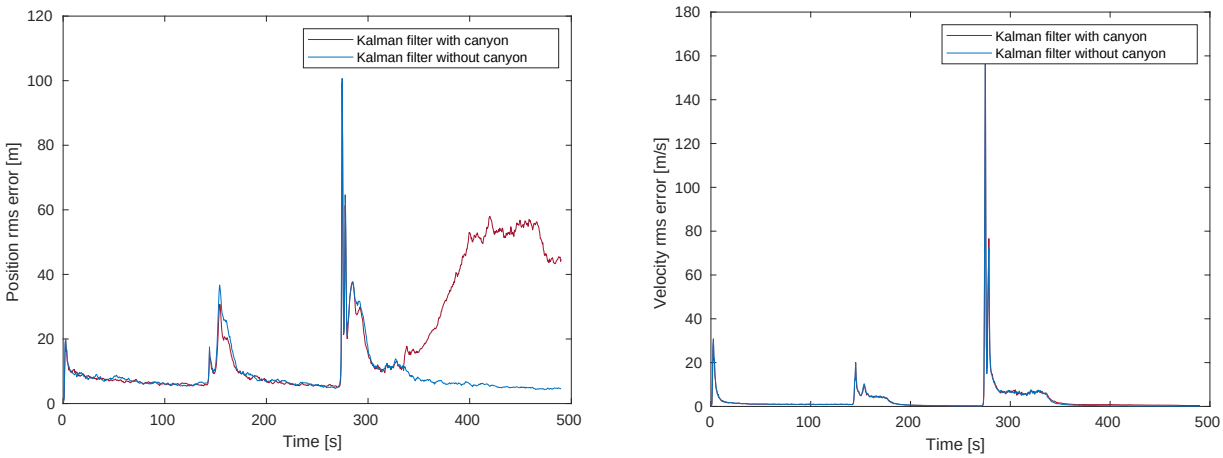
Constata-se pela Figura 8, que em geral o erro rms de posição obtido pelo EKF é inferior ao do *Least Mean Squares*, embora nas zonas de transição entre caminhos com características diferentes, existe um intervalo de tempo curto onde o Kalman Filter detém valores elevados de imprecisão, i.e. devido à mudança brusca que o EKF tem de se submeter. Em relação ao erro rms de velocidade, o EKF é realmente o que apresenta melhores resultados, exceto nas zonas e situações anteriormente referidas no erro rms de posição.



(a) v_x (b) v_y (c) v_z

Figura 9: Velocidades v_x, v_y e v_z determinadas pelo algoritmo LMS e KF, na presença de ruído não gaussiano e canyon

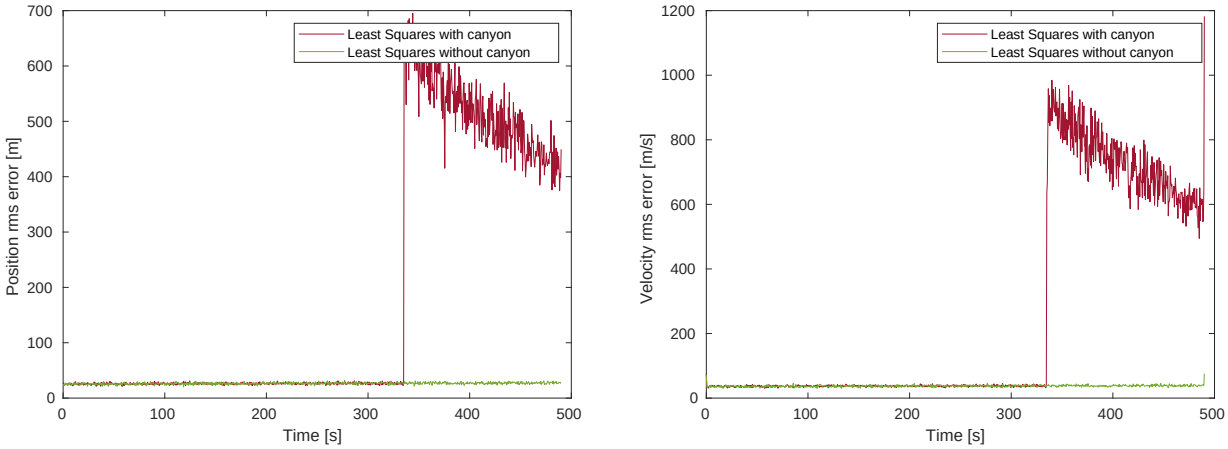
Verifica-se por observação dos resultados apresentados na Figura 9 a diferença de precisão entre o EKF e do LMS na determinação da velocidade. De facto, a determinação da velocidade v_x, v_y e v_z pelo *Least Mean Squares* é bastante afetada pela existência de ruído, principalmente na presença da situação de canyon.



(a) Erro rms da posição (b) Erro rms da velocidade

Figura 10: Comparação do erro rms da posição e velocidade no algoritmo KF aquando da aplicação da situação de canyon no caminho EF

Ao contrário do LMS, na situação de canyon, o EKF apresenta elevada robustez na precisão da posição e da velocidade, verificando-se que o efeito no erro rms de velocidade é reduzido comparativamente ao efeito que é sujeito o erro rms de posição à presença de canyon.

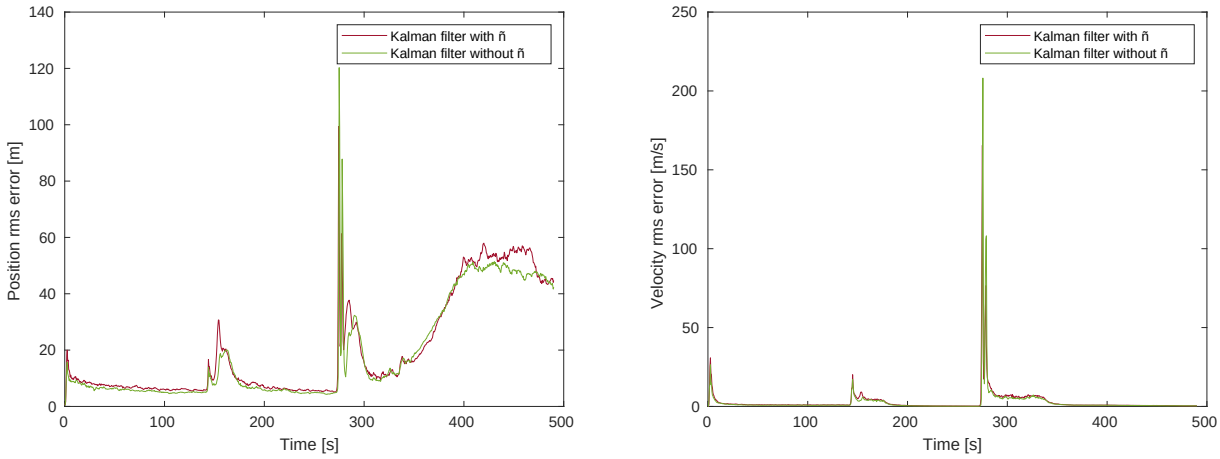


(a) Erro rms da posição (b) Erro rms da velocidade

Figura 11: Comparação do erro rms da posição e velocidade no algoritmo LMS aquando da aplicação da situação de canyon no caminho EF

Os gráficos anteriores demonstram o efeito excessivo que a situação de canyon implica no erro rms de posição e velocidade no *Least Squares*.

Apresenta-se de seguida uma analise do efeito da ionosfera nos resultados de precisão de posição e velocidade no EKF.



(a) Erro rms da posição (b) Erro rms da velocidade

Figura 12: Comparação do erro rms da posição e velocidade no algoritmo KF na presença e ausência de ruído ionosférico

Pelo dados presentes na Figura 12, pode-se afirmar que o efeito do ruído ionosférico tem pouco impacto na precisão do sistema de localização, pois verifica-se, quer no erro rms de posição, quer no erro rms de velocidade, diferenças que se poderiam considerar desprezáveis, embora o caso sem ruído ionosférico tem ligeiramente uma melhor precisão. Mais especificamente, o ruído ionosférico introduz um aumento no erro rms de posição de 1 m e no erro rms de velocidade 0.1 m/s.

9 Conclusão

Neste relatório verificou-se inicialmente a forma como as coordenadas obtidas no almanaque foram transformadas no referencial ECEF para calcular as pseudo-distâncias. Além deste referencial, também foi feita a análise do referencial local ENU, onde se conferiu que para que um satélite seja visível para um recetor local (sem qualquer outra condição) é necessário que a coordenada z do satélite neste referencial seja positiva. Posteriormente para a minimização do PDOP aplicou-se um algoritmo genético com intuito de resolver o problema de otimização discreto com baixo custo computacional.

Passando para a análise de resultados dos algoritmos, verificou-se que o EKF do modelo PV é mais robusto à presença do ruído do que o LMS, mas por causa disto também apresenta uma menor reatividade às variações de estado do sistema. Esta diferença de robustez comprova-se principalmente na situação de canyon. No que respeita aos erros rms, o EKF na maioria do tempo apresenta um erro de posição e velocidade inferior ao do LMS. Contudo, devido a mudança de trajetoria e à baixa reatividade do EKF existem intervalos em que este apresenta um erro muito maior. Mesmo assim, o EKF é o que apresenta melhores resultados no geral - é de salientar que as simulações apresentadas ao longo do relatório foram obtidas com q_v a variar, os resultados similares para q_v constante constam nos anexos. Outro aspeto a salientar

do EKF é que a presença da situação de canyon provoca um aumento significativo no erro de posição, enquanto que o erro da velocidade acaba por ser quase o mesmo. Por parte do LMS, a presença de canyon provoca, quer para posição, quer para velocidade, um aumento bastante grande do erro rms. Finalmente, constatou-se que o efeito do ruído da ionosfera acaba por ter pouco impacto na precisão do sistema, obtendo ligeiramente uma melhor precisão com a ausência deste.

Sendo assim, a partir dos resultados obtidos e analisados pode-se conferir que objetivos do projeto foram cumpridos e bem sucedidos.

Referências

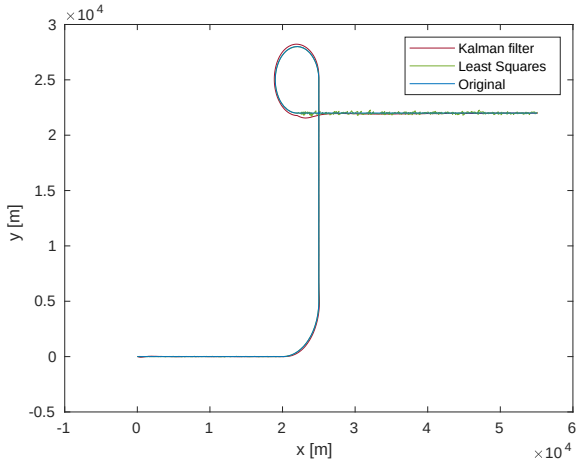
[1] Fernando Nunes. *Apontamentos de Sistemas de Controlo de Tráfego*. 2009.

10 Anexos

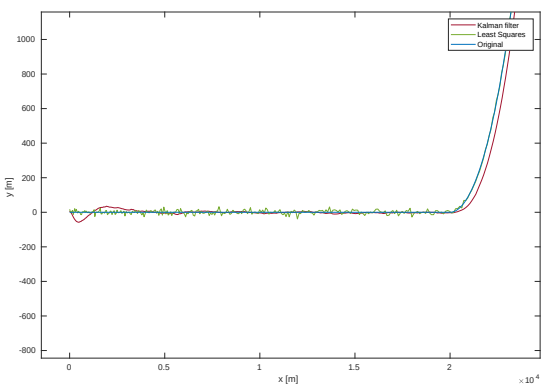
10.1 Resultados adicionais

10.1.1 Efeito de Q_k estático

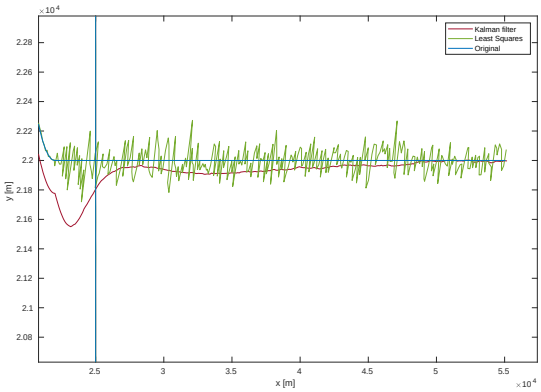
Para avaliar o efeito de utilização de Q_k estático, considerou-se $q_v = 0.1$ e obteve-se os seguintes resultados



(a) Trajetória total da simulação



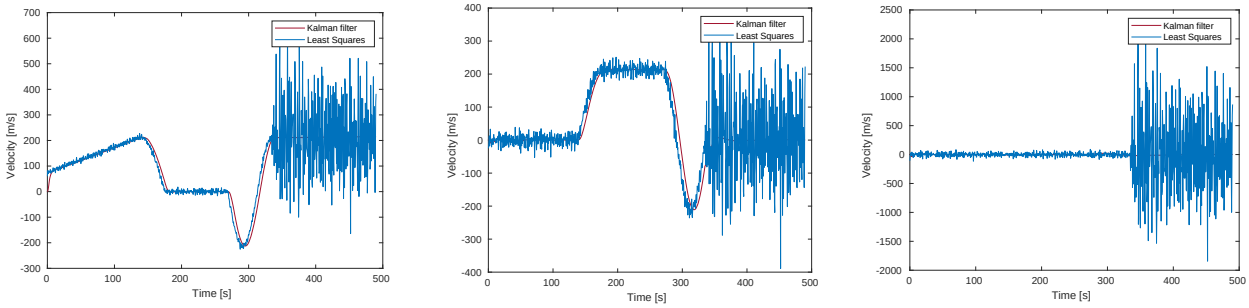
(b) Zoom da trajetória no caminho AB



(c) Zoom da trajetória no caminho EF

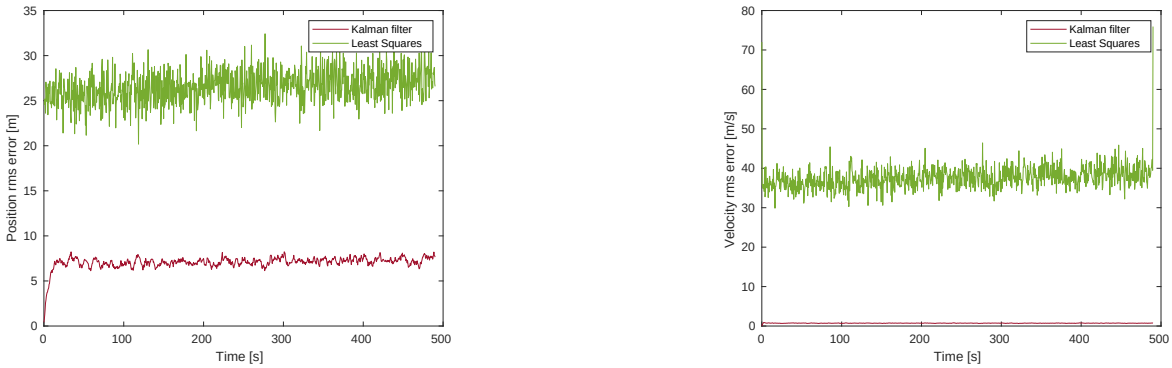
Figura 13: Comparação do comportamento das trajectórias - LMS, EKF com Q estático e trajetória original

Constata-se a utilização de Q_k estático implica problemas do algoritmo no seguimento de referência, principalmente nas zonas caraterizadas pelas curvas em velocidade linear constante.



(a) v_x (b) v_y (c) v_z

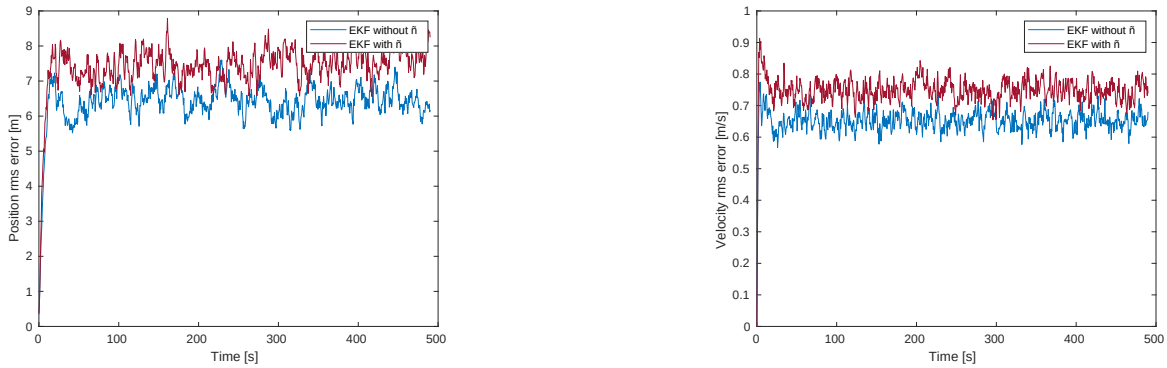
Figura 14: Velocidades em x, y, e z para os dois métodos - EKF (Q estático) e LMS



(a) Erro rms da posição (b) Erro rms da velocidade

Figura 15: Comparação do EKF com LMS com existência de ruído da ionosfera mas sem canyon.

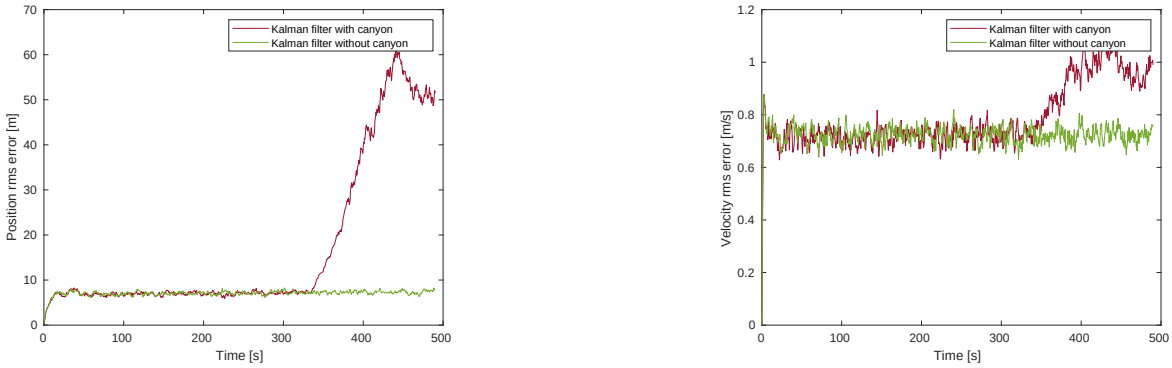
Salienta-se que ao contrário do EKF implementado no corpo principal deste relatório, o EKF com Q_k constante evita a ocorrência de picos elevados de rms na posição/velocidade aquando de variações de estado inesperadas pelo algoritmo.



(a) Erro rms da posição (b) Erro rms da velocidade

Figura 16: Efeito do ruído ionosférico no EKF quando se simula com Q_k constante, $q_v = 0.1$

Também neste caso o ruído ionosférico introduz um aumento no erro rms de posição e velocidade, verificando-se um aumento de 1 m no erro rms de posição e 0.1 m/s no erro rms de velocidade.

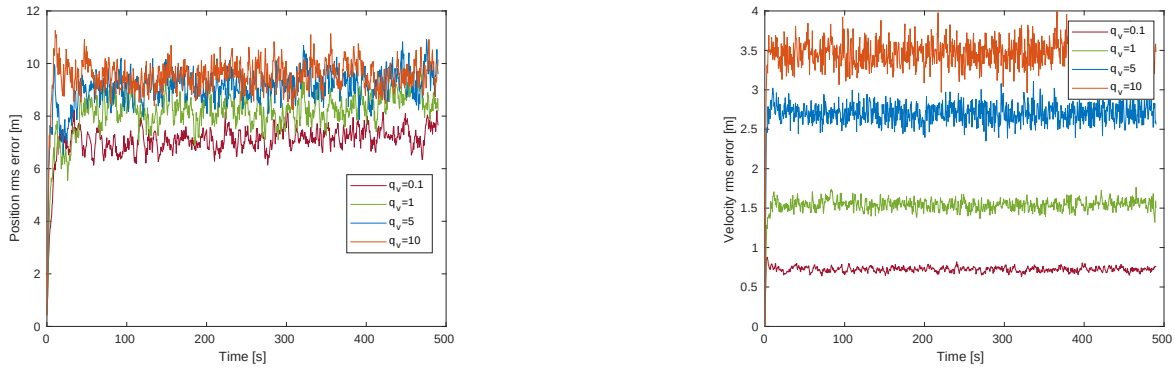


(a) Erro rms da posição (b) Erro rms da velocidade

Figura 17: Comparação da presença de canyon no EKF (Q estático)

Os anteriores resultados são semelhantes aos que já foram descritos na análise de resultados no corpo principal do relatório, i.e., os gráficos anteriores não apresentam mudanças significativas que devem ser comentadas, pois não apresentam informação adicional à já discutida no relatório.

Na Figura seguinte encontram-se resultados do erro rms de posição e velocidade para diferentes valores fixos de q_v.



(a) Erro rms da posição (b) Erro rms da velocidade

Figura 18: Comparação do erro rms de posição e velocidade para diferentes valores de q_v

Verifica-se que o aumento de q_v produz um aumento do erro rms de posição e velocidade, mas permite obter resultados mais exatos do algoritmo - este resultado associado à exatidão está apresentado na Figura 19.

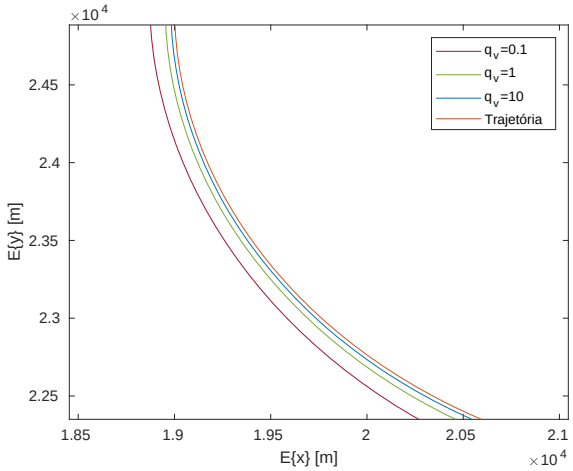
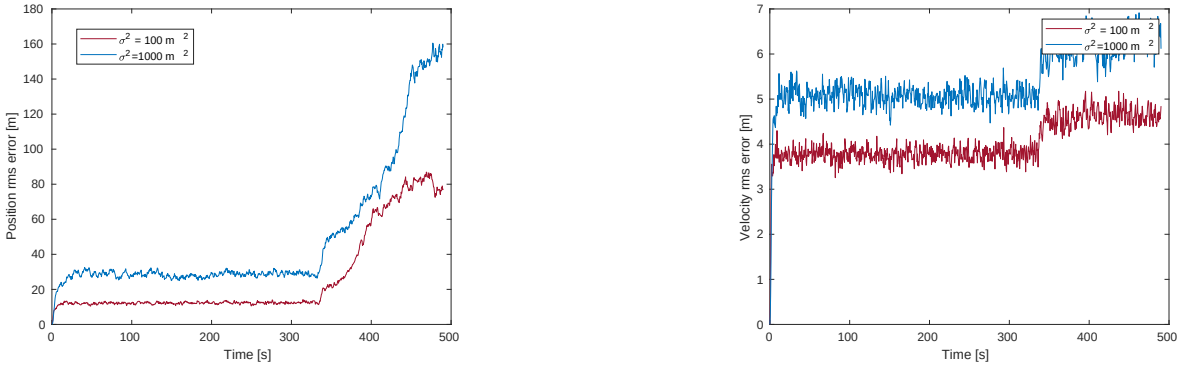


Figura 19: Resultados médios do EKF para diferentes valores fixos de q_v para um troço da trajetória

10.1.2 Efeito da variância do ruído

Apresenta-se de seguida resultados associados ao efeito causado na precisão da determinação da posição e velocidade para diferentes valores de σ^2.



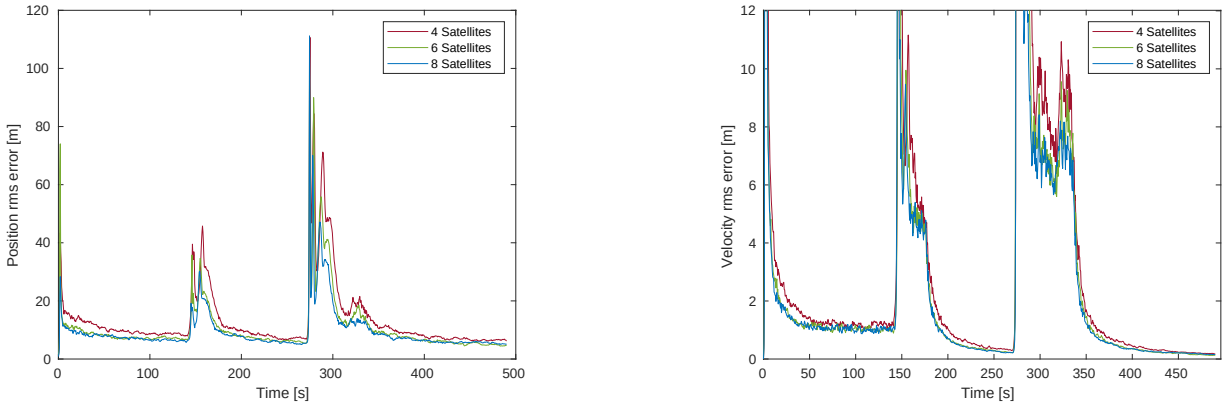
(a) Erro rms da posição (b) Erro rms da velocidade

Figura 20: Comparação do erro rms de posição e velocidade para diferentes valores de σ^2

Verifica-se pelos dados de simulação anteriores, $\sigma^2 = 100\ m^2$ e $\sigma^2 = 1000\ m^2$, que quanto maior o σ^2 maior é o erro rms de posição e velocidade, mais especificamente no caso apresentado, há um aumento de 10 m no erro rms de posição e 1 m/s no erro rms de velocidade.

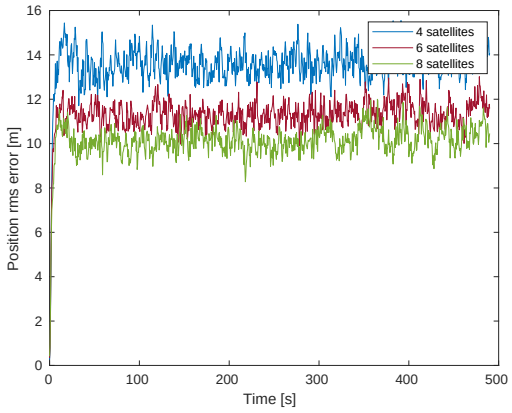
10.1.3 Efeito do número de satélites

Para avaliar o efeito que o número de satélites em utilização tem no EKF-PV, considerou-se 3 casos distintos, nomeadamente a utilização de 4, 6 e 8 satélites. Os resultados estão apresentados nas Figuras seguintes.

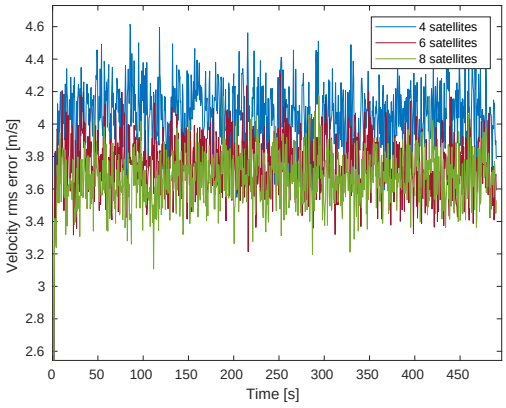


(a) Erro rms da posição (b) Erro rms da velocidade

Figura 21: Resultados obtidos do erro rms de posição e velocidade para diferentes números de satélites em utilização - os resultados estão associados ao caso onde q_v varia no tempo.



(a) Erro rms da posição



(b) Erro rms da velocidade

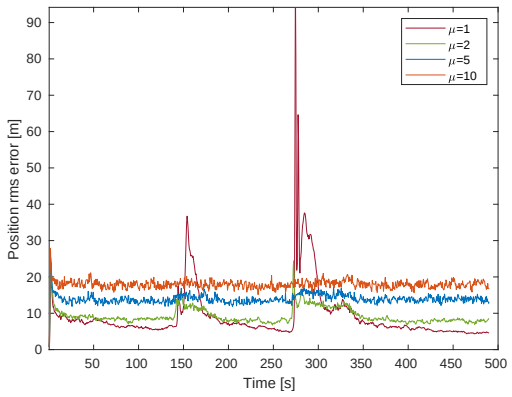
Figura 22: Resultados obtidos do erro rms de posição e velocidade para diferentes números de satélites em utilização - os resultados estão associados ao caso onde $q_v = 10$

Verifica-se que em geral, a utilização de mais satélites permite existir uma menor diluição de precisão na determinação da posição e velocidade.

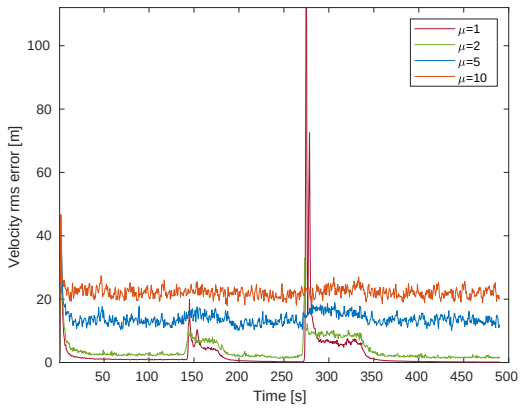
10.1.4 Efeito e ajuste do parâmetro μ

Nas secções principais do trabalho, observou-se a existência de picos nos erros rms de posição e velocidade, isso deveu-se à aproximação realizada para q_v. Considerou-se a utilização de um parâmetro μ na expressão (21) tal como apresentado na equação (35) - realizou-se várias iterações para este parâmetro e observou-se os resultados obtidos presentes na Figura 28.

$$q_{v_{i,k}} = \mu u_{i,k}^2 \tag{35}$$



(a) Erro rms da posição



(b) Erro rms da velocidade

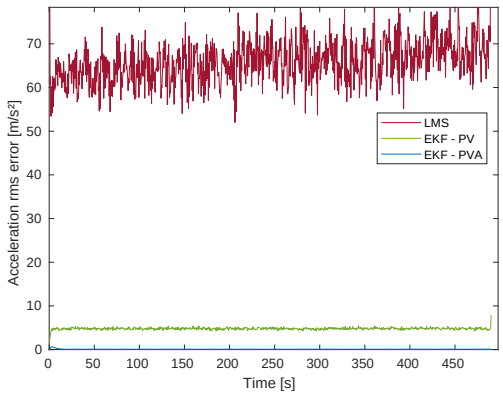
Figura 23: Resultados do ajuste do parâmetro μ

Com base nos resultados anteriores verifica-se que aumentar o valor de μ evita a existência de picos no erro rms de posição e velocidade na trajetória simulada, mas em contrapartida produz um aumento geral do erro rms. Para o caso em estudo, o melhor valor a considerar seria de $\mu = 2$.

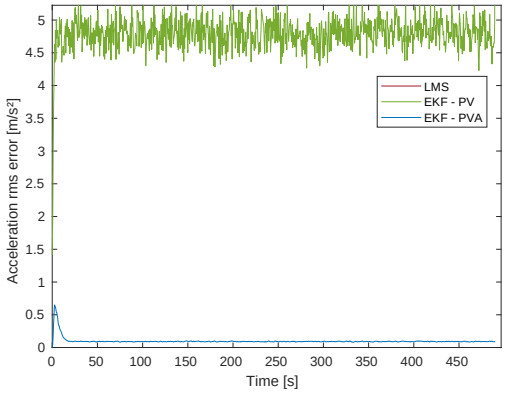
10.1.5 Comparação entre EKF-PV, EKF-PVA e LMS

Desenvolveu-se também o Kalman Filter generalizado com o modelo PVA e realizou-se a comparação de desempenho com o Kalman filter generalizado com modelo PV (EKF-PV) e com o Least Mean Squares (LMS). Nas figuras seguintes apresenta-se os resultados dos erros rms da aceleração, velocidade e posição, respetivamente.

Algumas especificações do EKF-PVA implementado e que está apresentado nos dados seguintes são: $\beta = 0.001$ e $\alpha = 0.1$



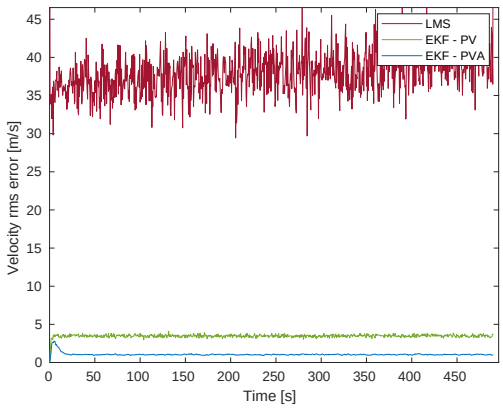
(a) Erro rms de aceleração



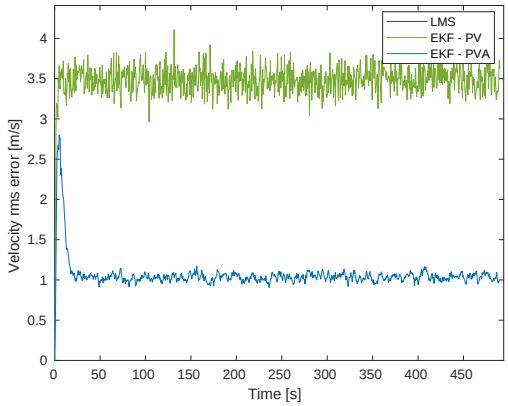
(b) Zoom do gráfico (a) para observar o EKF-PV e EKF-PVA

Figura 24: Erro rms da aceleração de LMS, EKF-PV e EKF-PVA

Verifica-se que na estimação da aceleração o EKF-PVA detém uma precisão superior aos outros observadores, em média o EKF-PVA apresenta um erro rms de aceleração de 0.1 m/s^2 enquanto que o EKF-PV ronda os 4.8 m/s^2 e o LMS ronda os 65 m/s^2.



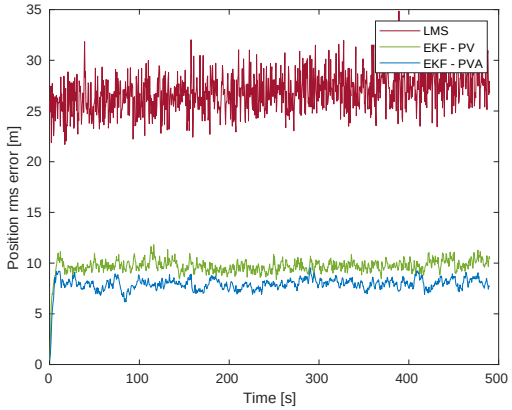
(a) Erro rms de velocidade



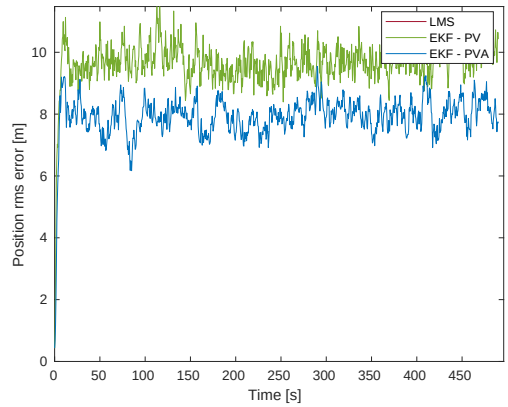
(b) Zoom do gráfico (a) para observar o EKF-PV e EKF-PVA

Figura 25: Erro rms da velocidade de LMS, EKF-PV e EKF-PVA

No caso da velocidade verifica-se o caso semelhante já apresentado na aceleração, o EKF-PVA é o que apresenta menor erro rms de velocidade, este valor ronda 1 m/s. Para o EKF-PV, o erro rms de velocidade é 3.5 m/s e para o LMS é 37 m/s.



(a) Erro rms de posição

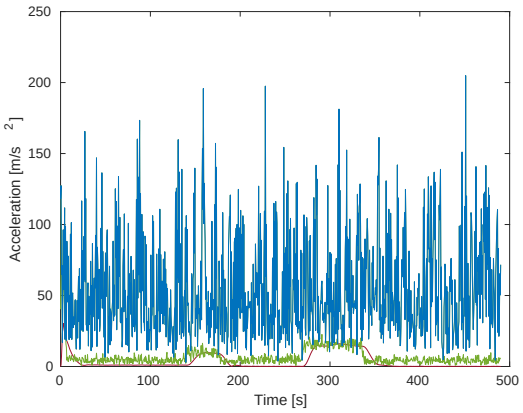


(b) Zoom do gráfico (a) para observar o EKF-PV e EKF-PVA

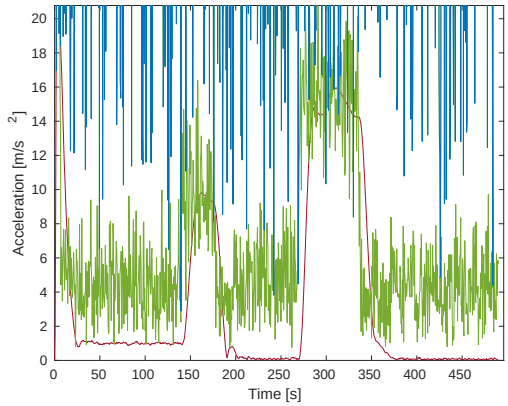
Figura 26: Erro rms da posição de LMS, EKF-PV e EKF-PVA

No caso do erro rms de posição, a discrepância de estimação entre o EKF-PV e o EKF-PVA já não é tão elevada, sendo que têm valores de erro rms de 8 m e 9 m, repetivamente. O LMS é o que apresenta pior desempenho, sendo o erro rms de posição de 26 m.

Para verificar a exatidão e o desempenho em uma iteração da simulação, apresenta-se os resultados do LMS, EKF-PV e EKF-PVA para a aceleração e velocidade ao longo da simulação.



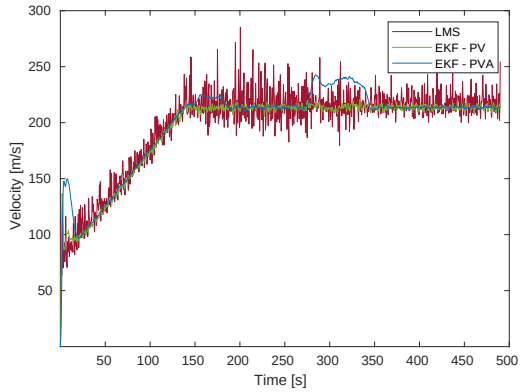
(a) Aceleração



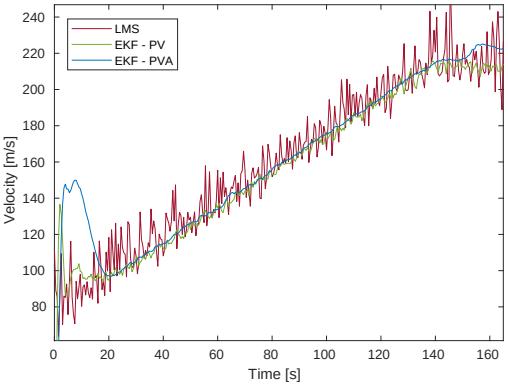
(b) Zoom do gráfico (a) para observar o EKF-PV e EKF-PVA

Figura 27: Resultados da aceleração ao longo da simulação de LMS, EKF-PV e EKF-PVA

Verifica-se que o observador que apresenta melhores resultados é o EKF-PVA, constatando-se que este é o menos afetado pelo ruído.



(a) Velocidade



(b) Zoom do gráfico (a)

Figura 28: Resultados da velocidade ao longo da simulação de LMS, EKF-PV e EKF-PVA

No caso da velocidade, verifica-se algo peculiar no EKF-PVA quando existe adição de ruído na simulação, a velocidade detém um erro constante na zona de curva.

10.2 Código do MATLAB

main.m

```
% SCT Project - Receiver positioning in GPS
% Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360

fprintf('-------------------------------------\n')
fprintf('%s \n BEGIN SIMULATION  \n',datetime)
tic;

global deg dt t sigma_eure
    sigma_eure = 7.1^2;
    deg=pi/180;
    dt=0.5;     % sampling time
    tfinal=490; % end time of simulation
    fprintf(' \t Sampling time = %.3f', dt)
    fprintf(' s \n \t End time = %.3f s \n', tfinal)

    flag4=1;    % flag4 = 0 -> Least squares
                % flag4 = 1 -> kalman filter PV model

for N=1:1 % nÃžmero de iteraÃğÃţes

    clearvars -except data_gps N deg dt t sigma_eure tfinal ...
         flag4 LS_iter kalman_iter

    fprintf(' \t iteraÃğÃčo %d \n', N)

    % auxiliar variables
    flag2=1; % for when to determine the new satellites to use
    flag3=1; % activate canyon scenario
    flag5=1; % if =1 use slow noise

    % 'PRN' -> struct with the info about the satellites
    PRN=leitura_sat_info();
    pos_A=referencia_A();

    % Create Dynamic matrix for PV model
    if flag4==1
        A_matrix = eye(8);
        for i=1:4
           A_matrix(i*2-1,i*2)=dt;
        end
    end

    V = 0;
    for t=0:dt:tfinal

        if t≥335.1878 && flag3==1 %canyon scenario
            sat=canyon_scenario(flag3, PRN, pos_A,sat);
```

```
48          end
49
50        if flag2==1
51          PRN=update_satellites_position(PRN,pos_A);
52          mask_angle=10; % angle in degrees
53          view_satellites=determine_view_satellites(PRN,pos_A,mask_angle);
54          flag=0;
55          number_of_satellites=4;
56          sat=minimize_PDOP(view_satellites,pos_A,flag,number_of_satellites);
57            flag2=0;
58        end
59
60        sat=update_satellites_position(sat,pos_A);
61
62        % geraÃğÃčo da trajetÃşria do recetor no plano horizontal estabelacido
63        % com a referencia A
64        [recetor.x,recetor.y]=trajectory();
65        aux=[[pos_A.ecef.x]; [pos_A.ecef.y];[pos_A.ecef.z]] + ...
66            inv(pos_A.ECEF_ENU)*[recetor.x;recetor.y;0];
67        recetor.ecef.x=aux(1); recetor.ecef.y=aux(2); recetor.ecef.z=aux(3);
68
69        [pseudoranges,pseudo_noise]=cal_pseudoranges_with_noise(...
70            sat,recetor,flag5);
71
72        % Condicoes iniciais
73        if t==0
74            if flag4==1
75              x = [recetor.ecef.x 0 recetor.ecef.y 0 recetor.ecef.z 0 0 0]';
76              anterior=x;
77              P = eye(8);
78            else
79                x = [recetor.ecef.x recetor.ecef.y recetor.ecef.z 0]';
80            end
81        end
82
83        if flag4==1
84            if flag5==1
85                 R = measurement_matrix([sat.elevation],sigma_eure);
86            else
87                 R= sigma_eure*eye(length(sat));
88            end
89            [z, H]= observation_matrix(x, sat);
90            K =kalman_value(P,H,R);
91            x = estimate_update(x, z, pseudo_noise', K);
92            P = covariance_update(P, K, H, R);
93            Q = create_Q2(anterior, x);
94            anterior=x;
95            [x,P] = prediction(A_matrix,x,P,Q);
96        else
97           [H,pseudorange_estimated]=LS_observation_matrix(x, sat);
98           Δ_x=inv(H'*H)*H'*(pseudo_noise' - pseudorange_estimated');
99           x=x + Δ_x;
```

```
        end

        % save data to plot in each instant
         V=V+1;
         if exist('data_gps')
             data_gps=save_data_plot(flag4, V, data_gps, pos_A, recetor, x);
         else
             data_gps=save_data_plot(flag4, V, [], pos_A, recetor, x);
         end
    end

    if flag4==0
        data_gps=LS_data_velocities(data_gps);
    end

    % dados para utilizar em processamento posterior (mean and var)
    if flag4==0
     LS_iter.x(N,:)=data_gps.LS.enu.x; LS_iter.vx(N,:)=data_gps.LS.enu.vx;
     LS_iter.y(N,:)=data_gps.LS.enu.y; LS_iter.vy(N,:)=data_gps.LS.enu.vy;
     LS_iter.z(N,:)=data_gps.LS.enu.z; LS_iter.vz(N,:)=data_gps.LS.enu.vz;
    else
     kalman_iter.x(N,:)=data_gps.kalman.enu.x;
     kalman_iter.y(N,:)=data_gps.kalman.enu.y;
     kalman_iter.z(N,:)=data_gps.kalman.enu.z;
     kalman_iter.vx(N,:)=data_gps.kalman.enu.vx;
     kalman_iter.vy(N,:)=data_gps.kalman.enu.vy;
     kalman_iter.vz(N,:)=data_gps.kalman.enu.vz;
    end

end
duration_simu=toc;
fprintf(' END OF SIMULATION \n \t Total duration = %.3f s \n',...
    duration_simu)
clear duration_simu;
fprintf('-------------------------------------\n')
```

leitura_sat_info.m

```
% SCT Project - Receiver positioning in GPS
% Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360

function PRN=leitura_sat_info()

fileID=fopen('almanac.txt','r');
aux=textscan(fileID,'%s','delimiter','\n');

j=1;
for i=1:15:length(aux{1,1})

    [aux4.ID]=sscanf(string(aux{1,1}(i+1)),'ID: %d');
    [aux4.health]=sscanf(string(aux{1,1}(i+2)),'Health:  %d');
```

```
14      [aux4.Eccentricity]=sscanf(string(aux{1,1}(i+3)),'Eccentricity: %f');
15      [aux4.Time]=sscanf(string(aux{1,1}(i+4)),'Time of Applicability(s): %f');
16      [aux4.Inclination]=sscanf(string(aux{1,1}(i+5)),'Orbital ...
           Inclination(rad):  %f');
17      [aux4.Rate_Ascen]=sscanf(string(aux{1,1}(i+6)),'Rate of Right ...
           Ascen(r/s): %f');
18      [aux4.SQRT]=sscanf(string(aux{1,1}(i+7)),'SQRT(A)  (m 1/2):  %f');
19      [aux4.Ascen]=sscanf(string(aux{1,1}(i+8)),'Right Ascen at Week(rad):  ...
           %f');
20      [aux4.Perigee]=sscanf(string(aux{1,1}(i+9)),'Argument of ...
           Perigee(rad):  %f');
21      [aux4.Anom]=sscanf(string(aux{1,1}(i+10)),'Mean Anom(rad):  %f');
22      [aux4.Af0]=sscanf(string(aux{1,1}(i+11)),'Af0(s):  %f');
23      [aux4.Af1]=sscanf(string(aux{1,1}(i+12)),'Af1(s/s):  %f');
24      [aux4.week]=sscanf(string(aux{1,1}(1+13)),'week: %d');
25
26      n=sqrt((3.986005*10^14)/(aux4.SQRT^6));
27      dt_aux=0;
28      M=aux4.Anom  + n*dt_aux;
29      syms E
30      eqn=M==E-aux4.Eccentricity*sin(E);
31      aux4.E=double(vpasolve(eqn,E));
32
33      % calculate argument of latitude, theta
34      aux4.theta=atan((sqrt(1 - aux4.Eccentricity^2)*sin(aux4.E))/...
35          (cos(aux4.E)-aux4.Eccentricity))+aux4.Perigee;
36
37      % calculate longitude of the ascending node, gamma
38      aux4.gamma=aux4.Ascen + aux4.Rate_Ascen*dt_aux - ...
39          (7.292115147*10^-5)*aux4.Time;
40
41      % calculate R
42      aux4.R=((aux4.SQRT)^2)*(1-aux4.Eccentricity*cos(aux4.E));
43
44      % calculate x, y & z
45      aux4.x=aux4.R*(cos(aux4.theta)*cos(aux4.gamma)-sin(aux4.theta)*...
46          sin(aux4.gamma)*cos(aux4.Inclination));
47      aux4.y=aux4.R*(cos(aux4.theta)*sin(aux4.gamma)+sin(aux4.theta)*...
48          cos(aux4.gamma)*cos(aux4.Inclination));
49      aux4.z=aux4.R*(sin(aux4.theta)*sin(aux4.Inclination));
50      % struct with the satellite info
51      PRN(j)=aux4;
52      j=j+1;
53  end
54      clear aux auxaxis equal4 n M
55      fclose(fileID);
56  end
```

referencia_A.m

```
1  % SCT Project - Receiver positioning in GPS
```

```
% Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360

function ref_A=referencia_A()

    ref_A.N=39;     % 39 degrees North - latitude of the receiver
    ref_A.W=8;      % 8 degrees West - longitude of the receiver
    ref_A.h=3500;   % h=3500m

    % determinar a posiÃğÃčo xyz do ponto A
    wgs84 = wgs84Ellipsoid;
    [ref_A.ecef.x,ref_A.ecef.y,ref_A.ecef.z] = geodetic2ecef(wgs84,...
        ref_A.N,ref_A.W,ref_A.h);

    ref_A.ECEF_ENU=[-sin(ref_A.W), cos(ref_A.W), 0;
    -sin(ref_A.N)*cos(ref_A.W), -sin(ref_A.N)*sin(ref_A.W), cos(ref_A.N);
    cos(ref_A.N)*cos(ref_A.W), cos(ref_A.N)*sin(ref_A.W), sin(ref_A.N)];

end
```

canyon_scenario.m

```
% SCT Project - Receiver positioning in GPS
% Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360

function sat=canyon_scenario(flag3, PRN, pos_A,sat)

     PRN=update_satellites_position(PRN,pos_A);
     aux2=[PRN.elevation];
     aux3=1:length(PRN);

     for i=1:4
        [M,I]=max(aux2);
        aux4(i)=aux3(I);
        aux2(I)=[];     aux3(I)=[];
     end
     clear sat
     for i=1:4
        sat(i)=PRN(aux4(i));
     end
     flag3=0;
end
```

update_satellites_position.m

```
% SCT Project - Receiver positioning in GPS
% Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360

function PRN=update_satellites_position(PRN,pos)

    global t
```

```
7       for i=1:length(PRN)
8           n=sqrt((3.986005*10^14)/(PRN(i).SQRT^6));
9           M=PRN(i).Anom + n*t;
10          E=0;
11          while true
12              A=E - (E - PRN(i).Eccentricity*sin(E) - M)/...
13                  (1 - PRN(i).Eccentricity*cos(E));
14             if abs(A-E)<0.001
15                 E=A;
16                 break
17             end
18             E=A;
19          end
20
21         PRN(i).E=E;
22
23         % calculate argument of latitude, theta
24         PRN(i).theta=atan((sqrt(1-PRN(i).Eccentricity^2)*sin(PRN(i).E))/...
25              (cos(PRN(i).E)-PRN(i).Eccentricity))+PRN(i).Perigee;
26
27          % calculate longitude of the ascending node, gamma
28          PRN(i).gamma=PRN(i).Ascen + PRN(i).Rate_Ascen*t - ...
29              (7.292115147*10^-5)*PRN(i).Time;
30
31          % calculate R
32          PRN(i).R=((PRN(i).SQRT)^2)*(1-PRN(i).Eccentricity*cos(PRN(i).E));
33
34          % calculate x, y & z
35          PRN(i).x=PRN(i).R*(cos(PRN(i).theta)*cos(PRN(i).gamma)-sin(...
36              PRN(i).theta)*sin(PRN(i).gamma)*cos(PRN(i).Inclination));
37          PRN(i).y=PRN(i).R*(cos(PRN(i).theta)*sin(PRN(i).gamma)+sin(...
38              PRN(i).theta)*cos(PRN(i).gamma)*cos(PRN(i).Inclination));
39          PRN(i).z=PRN(i).R*(sin(PRN(i).theta)*sin(PRN(i).Inclination));
40
41          PRN(i).enu=pos.ECEF_ENU*[PRN(i).x-pos.ecef.x;...
42              PRN(i).y-pos.ecef.y;PRN(i).z-pos.ecef.z];
43
44          PRN(i).elevation=asin(PRN(i).enu(3)/sqrt(PRN(i).enu(1)^2 + ...
45             PRN(i).enu(2)^2 + PRN(i).enu(3)^2));
46      end
47  end
```

determine_view_satellites.m

```
1   % SCT Project - Receiver positioning in GPS
2   % Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360
3
4   function [view_satellites]=determine_view_satellites(PRN,pos,mask_angle)
5       global deg
6
7       % enu: position relative to the reference position
```

```
8       for i=1:length(PRN)
9           PRN(i).enu=pos.ECEF_ENU*[PRN(i).x-pos.ecef.x;...
10              PRN(i).y-pos.ecef.y;PRN(i).z-pos.ecef.z];
11      end
12
13      % calculate the elevation of each satellite
14      j=1;
15      for i=1:length(PRN)
16         PRN(i).elevation=asin(PRN(i).enu(3)/sqrt(PRN(i).enu(1)^2 + ...
17             PRN(i).enu(2)^2 + PRN(i).enu(3)^2));
18         if PRN(i).elevation>mask_angle*deg
19              view_index(j)=i;
20              j=j+1;
21         end
22      end
23
24      % A structure only having the satellites in view
25      for i=1:length(view_index)
26         view_satellites(i)=PRN(view_index(i));
27      end
28
29  end
```

minimize_PDOP.m

```
1   % SCT Project - Receiver positioning in GPS
2   % Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360
3
4   function sat=minimize_PDOP(view_satellites, pos,flag ,number_of_satellites)
5
6       constraint(1)=flag;
7       constraint(2)=number_of_satellites;
8       IntCon = 1:length(view_satellites);
9       options = optimoptions('ga');
10      options = optimoptions(options,'MaxGenerations', 20);
11      options = optimoptions(options,'MaxStallGenerations', inf);
12      options = optimoptions(options,'FunctionTolerance', 0);
13      options = optimoptions(options,'ConstraintTolerance', 0);
14      options = optimoptions(options,'Display', 'off');
15      % options=optimoptions(options,'PlotFcn', { @gaplotbestf });
16      % options=optimoptions(options,'MutationFcn',{@mutationgaussian,1,.5});
17      [x,fval,exitflag,output,population,score] = ...
18      ga(@(x)cost_funct(x,view_satellites,pos, constraint),...
19      length(view_satellites),[],[],[],[], zeros(1,length(view_satellites)),...
20      ones(1,length(view_satellites)),[],IntCon,options);
21      % only get the satellites to work with
22      j=1;
23      for i=1:length(x)
24          if x(i)==1
25              sat(j)=view_satellites(i);
26              j=j+1;
```

```
        end
    end
end
```

cost_funct.m

```
% SCT Project - Receiver positioning in GPS
% Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360

function cost=cost_funct(on_off,view_satellites,pos, constraint)

    j=1;
    if sum(on_off)≥4
        for i=1:length(on_off)
          if on_off(i)==1
            r=sqrt((view_satellites(i).x-pos.ecef.x)^2 ...
               +(view_satellites(i).y ...
                - pos.ecef.y)^2 + (view_satellites(i).z - pos.ecef.z)^2);
            H(j,1)=(view_satellites(i).x - pos.ecef.x)/r;
            H(j,2)=(view_satellites(i).y ...
                - pos.ecef.y)/r;
            H(j,3)=(view_satellites(i).z - pos.ecef.z)/r;
            H(j,4)=-1;
            j=j+1;
          end
        end
        M_GDOP=inv(H'*H);
        PDOP=sqrt(M_GDOP(1,1) + M_GDOP(2,2) + M_GDOP(3,3));
        cost=PDOP;
    else
       cost=inf;
    end
    % if there is a constraint about the number of satellites to use
    if constraint(1)==1
        if sum(on_off)≠constraint(2)
            cost=inf;
        end
    end
end
```

trajectory.m

```
% SCT Project - Receiver positioning in GPS
% Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360

function [x,y]=trajectory()
    global t
v0=sqrt(2*20*10^3 + 75^2);
R1=5*10^3;
R2=3*10^3;
```

```
w0=v0/R1;
w1=v0/R2;
t1=25*73^(1/2) - 75;
t2=pi/(2*w0);
t3=(20*10^3)/v0;
t4=(6*pi)/(4*w1);
t5=(36*10^3)/v0;
% AB
if t≤ t1
    x=75*t + 0.5*t^2;
    y=0;
elseif t≤t1 + t2
% BCt5
    theta=w0*(t-t1);
    x=20*10^3 + R1*sin(theta);
    y=R1*(1-cos(theta));
elseif t≤t1 + t2 +t3
% CD
    y=R1 + v0*(t - t1 - t2);
    x=20*10^3 + R1;
elseif t≤t1 + t2 + t3 + t4
% DE
    theta=w1*(t -t1 - t2 - t3);
    x=(20*10^3 + R1) - R2*(1- cos(theta));
    y=(20*10^3 + R1) + R2*sin(theta);
elseif t≤t1 + t2 + t3 + t4 + t5
% EF
    x=(20*10^3  + R1 - R2) + v0*(t - t1 -t2 -t3 -t4);
    y=20*10^3  + R1 - R2;
elseif t>t1 + t2 + t3 + t4 + t5
    x=(20*10^3  + R1 - R2) + v0*(t5);
    y=20*10^3  + R1 - R2;
end

end
```

cal_pseudoranges_with_noise.m

```
% SCT Project - Receiver positioning in GPS
% Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360

function [pseudoranges,pseudoranges_plus_noise]=...
    cal_pseudoranges_with_noise(sat,recetor, flag5)
    global sigma_eure t

    c=(3*10^8);
    tu=10^-6; %atraso do relÃşgio
    for i=1:length(sat)
       pseudoranges(i)=sqrt((recetor.ecef.x - sat(i).x)^2  + ...
           (recetor.ecef.y-sat(i).y)^2  +(recetor.ecef.z-sat(i).z)^2)+c*tu;
       % create slowly varying noise
```

```
        if flag5==1
             slow_noise(i)=awgn(pseudoranges(i),-10*log10(sigma_eure/8));
             pseudoranges_plus_noise(i)= awgn(pseudoranges(i),-10*log10(...
                 7*sigma_eure/8)) + flag5*(slow_noise(i)- pseudoranges(i))...
                 *1.1/(sin(sat(i).elevation) + 0.1) ;
        else
             slow_noise(i)=0;
             pseudoranges_plus_noise(i)= awgn(pseudoranges(i),-10*log10(...
                 sigma_eure));
        end
    end
end
```

measurement_matrix.m

```
% SCT Project - Receiver positioning in GPS
% Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360

function R=measurement_matrix(elevation,desvio_padrao_squared)

for i=1:length(elevation)
    for j=1:length(elevation)
    if i==j
       R(i,j)=(desvio_padrao_squared/8)*(7 + ...
           1.21/(sin(elevation(i)) + 0.1)^2);
    else
        R(i,j)=(desvio_padrao_squared/8)*1.21/((sin(elevation(i))+ 0.1)...
            *(sin(elevation(j))) + 0.1);
    end
    end
end
end
```

observation_matrix.m

```
% SCT Project - Receiver positioning in GPS
% Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360

function [z, H]= observation_matrix(x, sat)
    % 8 por causa do modelo PV, caso for P ou PVA Ãl' diferente
    H = zeros(length(sat), 8);
    h = zeros(length(sat), 1);

    for i=1:length(sat)
        r_sat = sqrt((sat(i).x - x(1))^2 + (sat(i).y - x(3))^2 + ...
            (sat(i).z - x(5))^2);
        H(i,1) = -(sat(i).x - x(1))/r_sat;
        H(i,3) = -(sat(i).y - x(3))/r_sat;
        H(i,5) = -(sat(i).z - x(5))/r_sat;
        H(i,7) = 1;
```

```
15          h(i) = r_sat + x(7);
16      end
17      z = h;
18  end
```

kalman_value.m

```
1  % SCT Project - Receiver positioning in GPS
2  % Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360
3
4  function K=kalman_value(P,H,R)
5      K=P*H'*pinv(H*P*H' + R);
6  end
```

estimate_update.m

```
1  % SCT Project - Receiver positioning in GPS
2  % Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360
3
4  function x_upd = estimate_update(x, z, pseudoranges, K)
5      x_upd = x+ K*(pseudoranges - z);
6  end
```

covariance_update.m

```
1  % SCT Project - Receiver positioning in GPS
2  % Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360
3
4  function P_upd = covariance_update(P, K, H, R)
5
6  Id=eye(size(K,1), size(H,2));
7  P_upd = (Id-K*H)*P*(Id-K*H)'+ K*R*K';
8  end
```

create_Q2.m

```
1  % SCT Project - Receiver positioning in GPS
2  % Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360
3
4  function Q=create_Q2(anterior, x)
5
6      global dt
7      Q = zeros(8);
8      c=3e8;
9      q_phi = 2e-19/2; % quartz temperature-compensated oscillator
10     q_f = 2 * pi^2 * 2e-20;
11
12     for index = 1:2:6
13         Q(index, index) = ((x(index+1) - anterior(index+1))^2) * dt^3/3;
```

```
        Q(index, index+1) = ((x(index+1) - anterior(index+1))^2)*dt^2/2;
        Q(index+1, index) = Q(index, index+1);
        Q(index+1, index+1) = ((x(index+1) - anterior(index+1))^2)*dt;
    end
    Q(7,7)= (q_phi + q_f*dt^3/3)*c^2;
    Q(7,8) = (q_f*dt^2/2)*c^2;
    Q(8,7) = Q(7,8);
    Q(8,8) = q_f*dt*c^2;
end
```

prediction.m

```
% SCT Project - Receiver positioning in GPS
% Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360

function [x2,P2]=prediction(A,x1,P1,Q)
    x2=A*x1;
    P2=A*P1*A' + Q;
end
```

LS_observation_matrix.m

```
% SCT Project - Receiver positioning in GPS
% Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360

function [H,pseudorange_estimate]=LS_observation_matrix(x, sat)

    for i=1:length(sat)
      r_sat = sqrt((sat(i).x - x(1))^2 + (sat(i).y - x(2))^2 + ...
          (sat(i).z - x(3))^2);
      pseudorange_estimate(i)=r_sat;
      H(i,1) = -(sat(i).x - x(1))/r_sat;
      H(i,2) = -(sat(i).y - x(2))/r_sat;
      H(i,3) = -(sat(i).z - x(3))/r_sat;
      H(i,4) = 1;
    end
end
```

LS_data_velocities.m

```
% SCT Project - Receiver positioning in GPS
% Authors:  Renato Loureiro, 89708, Santiago Rodriguez, 90360

function data_gps=LS_data_velocities(data_gps)
    global dt
    data_gps.LS.ecef.vx = gradient(data_gps.LS.ecef.x,dt);
    data_gps.LS.ecef.vy = gradient(data_gps.LS.ecef.y,dt);
    data_gps.LS.ecef.vz = gradient(data_gps.LS.ecef.z,dt);
    data_gps.LS.enu.vx = gradient(data_gps.LS.enu.x,dt);
    data_gps.LS.enu.vy = gradient(data_gps.LS.enu.y,dt);
```

```
11      data_gps.LS.enu.vz = gradient(data_gps.LS.enu.z,dt);
12  end
```